Claims

1. A system for improving the parallelization of image processing, using one or more parallelization modes, wherein said image that is displayed on at least one computer screen by one or more Graphic Processing Units, comprising at least:
   a. one or more software applications, for issuing graphic commands;
   b. one or more graphic libraries, for storing data used to implement said graphic commands;
   c. one or more Software Hub Drivers, for controlling a Hardware Hub, for interacting with the operation system of said computer and said graphic libraries, for performing real-time analysis of a data stream, from which frames of said image are generated, for determining the parallelization mode of each GPU, and for forwarding said data stream or a portion thereof to each GPU;
   d. one or more GPU Drivers, for allowing said GPUs to interact with said graphic libraries; and
   e. at least one I/O module for interconnecting between said Software module and said Hardware Hub,

wherein, said Hardware Hub distributes, for each frame, between said GPUs, graphic commands and said data stream or a portion thereof, according to their relative complexity within said image, said complexity is defined by said Software Hub Driver; and composites a graphics output for display, using the outputs obtained from at least one GPU, while alternating, whenever required, said parallelization mode for said each frame.

2. System according to claim 1, wherein the parallelization is based on an object division mode or on an image division mode or on a time division mode or on any combination thereof.

3. System according to claim 1, wherein the hardware hub comprises at least one compositing unit at least for composing a complete frame from processed portions of the data stream.

4. System according to claim 2, wherein the hardware hub comprises at least one hub router for routing polygonal data, for routing graphic command stream, for routing pixel data and for routing the results of composition, while operating in the object division mode or in the image division mode or in the time division mode or in any combination thereof.

5. System according to claim 1, wherein the hardware hub comprises at least one control unit for receiving commands from the Software Hub Driver within the I/O module.

6. System according to claim 1, wherein the hardware hub comprises a memory unit for storing intermediate processing results of one or more GPUs and data required for composition and transferring the processed data for display.

7. System according to claim 1, wherein the Software Hub Driver is capable of performing the following operations:
    a. interception of the graphic commands from the standard graphic library by means of the OS interface and utilities;
    b. forwarding and creating graphic commands to the GPU Driver by means of the OS interface and utilities;
    c. controlling the Hardware Hub, registry and installation operations by means of the OS interface and utilities:
    d. maintaining the consistency of graphic machine states across the GPUs, based on the input graphic commands stream, while using state monitoring;

e. estimating the type of graphic load and overload in the executed application graphic context, while using application and graphic resources analysis;
f. load estimation of the GPUs load balance based on graphic commands stream and time measurements, while using application and graphic resources analysis;
g. adjusting the load distribution between GPUs according to feedback received from each GPU regarding said load balance, while using application and graphic resources analysis;
h. performing manipulation in graphic functions according to the current parallelization mode; and
i. controlling the distributed graphic functions, while modifying said graphic commands and said data stream according to said current parallelization mode.

8. A method for improving the parallelization of image processing, using one or more parallelization modes, wherein said image that is displayed on at least one computer screen by one or more Graphic Processing Units, comprising at least:
   a. providing one or more software applications, for issuing graphic commands;
   b. providing one or more graphic libraries, for storing data used to implement said graphic commands;
   c. providing one or more Software Hub Drivers, for controlling a Hardware Hub, for interacting with the operation system of said computer and said graphic libraries, for performing real-time analysis of a data stream, from which frames of said image are generated, for determining the parallelization mode of each GPU, and for forwarding said data stream or a portion thereof to each GPU;

d. providing one or more GPU Drivers, for allowing said GPUs to interact with said graphic libraries.
e. providing at least one I/O module for interconnecting between said Software module and said Hardware Hub; and
f. for each frame, distributing between said GPUs and by means of said Hardware Hub, graphic commands and said data stream or a portion thereof, according to their relative complexity within said image, said complexity is defined by said Software Hub Driver; and compositing a graphics output for display, using the outputs obtained from at least one GPU, while alternating, whenever required, said parallelization mode for said each frame.

9. A method according to claim 8, wherein the parallelization mode is an Object division parallelization mode and the following steps are performed:
   a. for each frame, generating a stream of graphic operations and polygonal data;
   b. marking the polygonal data and graphic commands by means of the Software Hub Driver for distribution between multiple GPUs;
   c. sending the marked data to the Hardware Hub;
   d. distributing said marked data via the Hub Router to said multiple GPUs;
   e. rendering the data by means of GPUs;
   f. retrieving the data from the Frame Buffers and forwarding the retrieved data to the compositing unit via the Hub Router;
   g. compositing the content of said Frame Buffers into a single Frame Buffer; and
   h. forwarding the content of said single Frame Buffer to at least one designated GPU for display.

10. A method according to claim 8, wherein the parallelization mode is an Image division parallelization mode and the following steps are performed:
    a. subdividing the screen to portions and assigning different viewports to GPUs by means of the Software Hub Driver;
    b. moving the entire polygonal data and graphic commands to the Hub Router;
    c. transmitting said entire polygonal data and graphic commands to GPUs, wherein each GPU receives the same data;
    d. rendering the data by means of GPUs;
    e. forwarding a portion of the content stored in the Frame Buffers to compositing unit in Hardware Hub for the complete image creation; and
    f. forwarding said image to at least one designated GPU for display.

11. A method according to claim 8, wherein the parallelization mode is a Time division parallelization mode and the following steps are performed:
    a. forwarding to each one of the multiple GPUs the entire amount of polygons for rendering;
    b. redirecting the entire polygonal data and graphic commands by means of Software Hub Driver to all GPUs, while alternating between them;
    c. rendering the data by means of GPUs;
    d. transferring rendered data from at least one GPU via the Hub Router; and
    e. redirecting the resulting content of the Frame Buffer via Hub Router to at least one designated GPU for display.

12. A method according to claim 8, wherein the distribution of polygons between multiple GPUs is performed by:
    a. distributing blocks of data between multiple GPUs;

b. testing each graphic operation for blocking mode, in which one or more parallelization modes are carried out;
c. redirecting the data in regular non-blocking path to at least one designated GPU;
d. repeating step (b) and (c) until a blocking operation is detected;
e. synchronizing GPUs by the following sequence:
    e.1. performing a flush operation in order to terminate rendering and clean up the internal pipeline in each GPU;
    e.2. performing a composition operation for merging the contents of the Frame Buffers into a single Frame Buffer; and
    e.3. transmitting said single Frame Buffer back to all GPUs;
f. terminating the composited complete frame at all GPUs, except one or more designated GPUs, whenever a Swap operation is detected and displaying the image by means of said one or more designated GPUs;
g. processing the same data by means of all GPUs, as long as the blocking mode is active and the Swap operation is not detected; and
h. continuing to process the designated data by means of multiple GPUs, whenever the blocking mode is inactive.

13. Method according to claim 8, wherein the parallelization is based on an object division mode or on an image division mode or on a time division mode or on any combination thereof.